Refine Search

Search Results -

Terms	Documents
(finite state machine) with (writ$3 near4 (once or one time))	0

Database: US Pre-Grant Publication Full-Text Database; US Patents Full-Text Database; US OCR Full-Text Database; EPO Abstracts Database; JPO Abstracts Database; Derwent World Patents Index; IBM Technical Disclosure Bulletins

Search:

Refine Search | Recall Text | Clear | Interrupt

Search History

DATE: Tuesday, October 25, 2005 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ			
L8	(finite state machine) with (writ$3 near4 (once or one time))	0	L8
L7	(finite state machine) with (idle or wait$3)	9	L7
DB=PGPB,USPT; PLUR=YES; OP=ADJ			
L6	L2 and l1	3	L6
L5	(finite state machine) with (writ$3 near4 (once or one time))	0	L5
L4	(finite state machine) with (writ$3 adj once)	0	L4
L3	(finite state machine) with (writ$3 adj2 (twice or two times))	0	L3
L2	(finite state machine) with (writ$3 and once)	19	L2
L1	(finite state machine) with (idle or wait$3)	140	L1

END OF SEARCH HISTORY

Results for "**((finite state machine and ((idle and wait$) <near/4> (state or mode)) and write once and writ...**" e-mail
Your search matched **0** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

» Search Options

View Session History

New Search

Modify Search

((finite state machine and ((idle and wait$) <near/4> (state or mode)) and write once a

Check to search only within this results set

Display Format: Citation Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

No results were found.

Please edit your search criteria and try again. Refer to the Help pages if you need assistan search.

Terms used
read and **writ** and **finite** **state** **machine** and **idle** and **wait** **near/4** **state** or **mode** and **write** **once** and **write** **twice**

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 The family of concurrent logic programming languages

Ehud Shapiro

September 1989 **ACM Computing Surveys (CSUR)**, Volume 21 Issue 3

Publisher: ACM Press

Full text available: pdf(9.62 MB) Additional Information: full citation, abstract, references, citings, index terms

Concurrent logic languages are high-level programming languages for parallel and distributed systems that offer a wide range of both known and novel concurrent programming techniques. Being logic programming languages, they preserve many advantages of the abstract logic programming model, including the logical reading of programs and computations, the convenier of representing data structures with logical terms and manipulating them using unification, and amenability to metaprogrammin ...

2 System-level power optimization: techniques and tools

Luca Benini, Giovanni de Micheli

April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volun Issue 2

Publisher: ACM Press

Full text available: pdf(385.22 KB) Additional Information: full citation, abstract, references, citings, index terms

This tutorial surveys design methods for energy-efficient system-level design. We consider electronic sytems consisting of a hardware platform and software layers. We consider the three major constituents of hardware that consume energy, namely computation, communication, an storage units, and we review methods of reducing their energy consumption. We also study models for analyzing the energy cost of software, and methods for energy-efficient software design and compilation. This survery ...

3 Parallel execution of prolog programs: a survey

Gopal Gupta, Enrico Pontelli, Khayri A.M. Ali, Mats Carlsson, Manuel V. Hermenegildo

July 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 2: Issue 4

Publisher: ACM Press

Full text available: pdf(1.95 MB) Additional Information: full citation, abstract, references, citings, index terms

Since the early days of logic programming, researchers in the field realized the potential for exploitation of parallelism present in the execution of logic programs. Their high-level nature, th presence of nondeterminism, and their referential transparency, among other characteristics, make logic programs interesting candidates for obtaining speedups through parallel execution. A the same time, the fact that the typical applications of logic programming frequently involve irregular computatio ...

Keywords: Automatic parallelization, constraint programming, logic programming, parallelism, prolog

4 Pipeline Architecture

C. V. Ramamoorthy, H. F. Li

January 1977 **ACM Computing Surveys (CSUR)**, Volume 9 Issue 1

Publisher: ACM Press

Full text available: pdf(3.53 MB) Additional Information: full citation, references, citings, index terms

5 On randomization in sequential and distributed algorithms

Rajiv Gupta, Scott A. Smolka, Shaji Bhaskar

March 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 1

Publisher: ACM Press

Full text available: pdf(8.01 MB) Additional Information: full citation, abstract, references, citings, index terms

Probabilistic, or randomized, algorithms are fast becoming as commonplace as conventional deterministic algorithms. This survey presents five techniques that have been widely used in the design of randomized algorithms. These techniques are illustrated using 12 randomized algorithms—both sequential and distributed— that span a wide range of applications, including:primality testing (a classical problem in number theory), interactive probabilistic proof s ...

Keywords: Byzantine agreement, CSP, analysis of algorithms, computational complexity, dinin philosophers problem, distributed algorithms, graph isomorphism, hashing, interactive probabili proof systems, leader election, message routing, nearest-neighbors problem, perfect hashing, primality testing, probabilistic techniques, randomized or probabilistic algorithms, randomized quicksort, sequential algorithms, transitive tournaments, universal hashing

6 Compiler-based I/O prefetching for out-of-core applications

Angela Demke Brown, Todd C. Mowry, Orran Krieger

May 2001 **ACM Transactions on Computer Systems (TOCS)**, Volume 19 Issue 2

Publisher: ACM Press

Full text available: pdf(499.03 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Current operating systems offer poor performance when a numeric application's working set do not fit in main memory. As a result, programmers who wish to solve "out-of-core" problems efficiently are typically faced with the onerous task of rewriting an application to use explicit I/C operations (e.g., read/write). In this paper, we propose and evaluate a fully automatic techniqu which liberates the programmer from this task, provides high performance, and requires only minima ...

Keywords: compiler optimization, prefetching, virtual memory

7 Distributed systems - programming and management: On remote procedure call

Patrícia Gomes Soares

November 1992 **Proceedings of the 1992 conference of the Centre for Advanced Studies on Collaborative research - Volume 2**

Publisher: IBM Press

Full text available: pdf(4.52 MB) Additional Information: full citation, abstract, references, citings

The Remote Procedure Call (RPC) paradigm is reviewed. The concept is described, along with th backbone structure of the mechanisms that support it. An overview of works in supporting thes mechanisms is discussed. Extensions to the paradigm that have been proposed to enlarge its suitability, are studied. The main contributions of this paper are a standard view and classificati of RPC mechanisms according to different perspectives, and a snapshot of the paradigm in use today and of goals for t ...

8 Abstract state machines capture parallel algorithms

Andreas Blass, Yuri Gurevich

October 2003 **ACM Transactions on Computational Logic (TOCL)**, Volume 4 Issue 4

Publisher: ACM Press

Full text available: pdf(610.28 KB) Additional Information: full citation, abstract, references, index terms

We give an axiomatic description of parallel, synchronous algorithms. Our main result is that ev such algorithm can be simulated, step for step, by an abstract state machine with a backgrounc that provides for multisets.

Keywords: ASM thesis, Parallel algorithm, abstract state machine, postulates for parallel computation

9 SODA: a simplified operating system for distributed applications

Jonathan Kepecs, Marvin Solomon
October 1985 **ACM SIGOPS Operating Systems Review**, Volume 19 Issue 4
Publisher: ACM Press
Full text available: pdf(1.05 MB) Additional Information: full citation, references, citings

10 SODA: A simplified operating system for distributed applications

Jonathan Kepecs, Marvin Solomon
August 1984 **Proceedings of the third annual ACM symposium on Principles of distributed computing**
Publisher: ACM Press
Full text available: pdf(925.18 KB) Additional Information: full citation, abstract, references, citings, index terms

The design and implementation study discussed in this paper can be viewed in two ways. On on hand, it represents a contribution to the active area of design of "smart" communications controllers which use increasingly sophisticated processor/memory configurations to improve th performance of interprocessor communication. On the other hand, it represents an application c the minimalist principles of the RISC (Reduced Instruction-Set Computer) architecture [1] to operating syst ...

11 Parallel RAMs with owned global memory and deterministic context-free language recognition

Patrick W. Dymond, Walter L. Ruzzo
January 2000 **Journal of the ACM (JACM)**, Volume 47 Issue 1
Publisher: ACM Press
Full text available: pdf(223.64 KB) Additional Information: full citation, abstract, references, index terms, review

We identify and study a natural and frequently occurring subclass of Concurrent Read, Exclusive Write Parallel Random Access Machines (CREW-PRAMs). Called Concurrent Read, Owner Write, CROW-PRAMS, these are machines in which each global memory location is assigned a unique "owner" processor, which is the only processor allowed to write into it. Considering the difficulti that would be involved in physically realizinga full CREW-PRAM model and demonstrate i ...

Keywords: CROW-PRAM, DCFL recognition, owner write, parallel algorithms

12 A Survey of Some Theoretical Aspects of Multiprocessing

J. L. Baer
January 1973 **ACM Computing Surveys (CSUR)**, Volume 5 Issue 1
Publisher: ACM Press
Full text available: pdf(4.05 MB) Additional Information: full citation, references, citings, index terms

13 Techniques for reducing consistency-related communication in distributed shared-memory systems

John B. Carter, John K. Bennett, Willy Zwaenepoel
August 1995 **ACM Transactions on Computer Systems (TOCS)**, Volume 13 Issue 3
Publisher: ACM Press
Full text available: pdf(2.86 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Distributed shared memory (DSM) is an abstraction of shared memory on a distributed-memory machine. Hardware DSM systems support this abstraction at the architecture level; software DS systems support the abstraction within the runtime system. One of the key problems in building efficient software DSM system is to reduce the amount of communication needed to keep the distributed memories consistent. In this article we present four techniques for doing so: softwar release consistency; m ...

Keywords: cache consistency protocols, distributed shared memory, memory models, release consistency, virtual shared memory

14 Experience with transactions in QuickSilver

Frank Schmuck, Jim Wylie

September 1991 **ACM SIGOPS Operating Systems Review , Proceedings of the thirteenth AC symposium on Operating systems principles**, Volume 25 Issue 5

Publisher: ACM Press , ACM Press

Full text available: pdf(1.66 MB) Additional Information: full citation, abstract, references, citings, index terms

All programs in the QuickSilver distributed system behave atomically with respect to their updat to permanent data. Operating system support for *transactions* provides the framework required support this, as well as a mechanism that unifies reclamation of resources after failures or norm process termination. This paper evaluates the use of transactions for these purposes in a gener purpose operating system and presents some of the lessons learned from our experience with a complet ...

15 Adaptive History-Based Memory Schedulers

Ibrahim Hur, Calvin Lin

December 2004 **Proceedings of the 37th annual International Symposium on Microarchitecture**

Publisher: IEEE Computer Society

Full text available: pdf(220.26 KB) Additional Information: full citation, abstract

As memory performance becomes increasingly important to overall system performance, the ne to carefully schedule memory operations also increases. This paper presents a new approach to memory scheduling that considers the history of recently scheduled operations. This history-bas approach provides two conceptual advantages: (1) it allows the scheduler to better reason abou the delays associated with its scheduling decisions, and (2) it allows the scheduler to select operations so that they ...

16 Data-Driven and Demand-Driven Computer Architecture

Philip C. Treleaven, David R. Brownbridge, Richard P. Hopkins

January 1982 **ACM Computing Surveys (CSUR)**, Volume 14 Issue 1

Publisher: ACM Press

Full text available: pdf(4.14 MB) Additional Information: full citation, references, citings, index terms

17 "Topologies"—distributed objects on multicomputers

Karsten Schwan, Win Bo

May 1990 **ACM Transactions on Computer Systems (TOCS)**, Volume 8 Issue 2

Publisher: ACM Press

Full text available: pdf(3.83 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Application programs written for large-scale multicomputers with interconnection structures kno to the programmer (e.g., hypercubes or meshes) use complex communication structures for connecting the applications' parallel tasks. Such structures implement a wide variety of function including the exchange of data or control information relevant to the task computations and/or communications required for task synchronization, message forwarding/filtering under program control, and so o ...

18 The embedded machine: predictable, portable real-time code

Thomas A. Henzinger, Christoph M. Kirsch

May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2002 Conference on**

Programming language design and implementation, Volume 37 Issue 5
Publisher: ACM Press , ACM Press
Full text available: pdf(223.85 KB) Additional Information: full citation, abstract, references, citings, index terms

The Embedded Machine is a virtual machine that mediates in real time the interaction between software processes and physical processes. It separates the compilation of embedded programs into two phases. The first, platform-independent compiler phase generates E code (code execut by the Embedded Machine), which supervises the timing ---not the scheduling--- of application tasks relative to external events, such as clock ticks and sensor interrupts. E~code is portable a exhibits, given an inpu ...

Keywords: real time, virtual machine

19 Comparative evaluation of latency reducing and tolerating techniques
Anoop Gupta, John Hennessy, Kourosh Gharachorloo, Todd Mowry, Wolf-Dietrich Weber
April 1991 **ACM SIGARCH Computer Architecture News , Proceedings of the 18th annual international symposium on Computer architecture**, Volume 19 Issue 3
Publisher: ACM Press , ACM Press
Full text available: pdf(1.36 MB) Additional Information: full citation, references, citings, index terms

20 Cache Refill/Access Decoupling for Vector Machines
Christopher Batten, Ronny Krashinsky, Steve Gerding, Krste Asanovic
December 2004 **Proceedings of the 37th annual International Symposium on Microarchitecture**
Publisher: IEEE Computer Society
Full text available: pdf(319.32 KB) Additional Information: full citation, abstract

Vector processors often use a cache to exploit temporal locality and reduce memory bandwidth demands, but then require expensive logic to track large numbers of outstanding cache misses sustain peak bandwidth from memory. We present refill/access decoupling, which augments the vector processor with a Vector Refill Unit (VRU) to quickly pre-execute vector memory comman and issue any needed cache line refills ahead of regular execution. The VRU reduces costs by eliminating much of the outstan ...